The priority of the commanded effect is compared to the priorities of the loaded effects; the first effect having a lower priority is eligible to be swapped with the commanded effect. Alternatively, all the loaded effects can be examined, and the effect having the lowest priority can be eligible to be replaced with the commanded effect if the commanded effect has a higher priority than that effect. In some embodiments, only effects not currently playing are examined for availability; alternatively, all the loaded effects, whether currently playing or not, can be examined and the lowest priority effect unloaded.

Furthermore, in some embodiments the priorities of effects for caching purposes can be changed by an operating system, application, or other program or user. For example, a developer of a force feedback application program can in some embodiments assign priorities to particular effects so that the developer has the flexibility to designate the importance of various effects to his or her particular application program. A priority system for a particular application could be provided to the host driver upon start-up of the application program. Such a priority system could be stored in a context for that application program, as described for Fig. 2, for example. In such a system, the developer should be able to assign the highest possible priority to any effect desired, which will cause a commanded effect having such a priority to be always loaded on the device regardless of which effects are already loaded. This allows the application to directly command force feedback on the device with no concerns about receiving failure messages.

In addition, effects can be organized into various categories or "suites", where the effects in a category are assigned priorities and/or where only particular categories need be in use at a particular time. This allows effects from other "inactive" categories to be unloaded from the device and effects included in the "active" category to be loaded. The priorities in some cases can be assigned by the developer of an application program. For example, a developer of a game application can make a category "On Land" which includes a collision effect and a weapon fire effect as priority 1, an engine rumble effect as priority 2, and a "slight breeze" effect as priority 3. The developer also can make a category of "In Water" including a water resistance (damping) effect and explosion effect as priority 1, a "strong current" effect as priority 2, and "hitting sea kelp" as priority 4. The application program calls the API to inform the host driver which category is currently in use, and when to switch categories. When, in the game, the user controls a vehicle to move from land into water, the application program indicates that the "On Land" category of effects should be switched to the "In Water" category of effects. The host driver then knows that all "On Land" effects are free to be unloaded from the device memory and that the "In Water" effects should be loaded. Furthermore, since each effect has been assigned a priority, the host driver knows that if there is not enough slots to store all of the "On Water" effects, the water resistance and explosion effects should be loaded before the lower priority effects. Other uses of categories and priorities is described in co-pending patent application serial no. 09/306,002

(TH.)
09/20/07